Can we Measure Computational Thinking with Tools? Present and Future of Dr. Scratch

Jesús Moreno-León
Programamos & Universidad Rey Juan Carlos
Seville, Spain
jesus.moreno@programamos.es

Gregorio Robles
Universidad Rey Juan Carlos
Madrid, Spain
grex@gsyc.urjc.es

Marcos Román-González
Universidad Nacional de Educación a Distancia
Madrid, Spain
mroman@edu.uned.es

Abstract

Dr. Scratch is a web-tool that analyzes Scratch projects to assess the development of computational thinking skills. This paper presents the current state of the validation process of the tool. The process involves several investigations to test the validity of Dr. Scratch from different perspectives, such as the extent to which learners improve their coding skills while using the tool in real life scenarios; the relationships of the score provided by the tool with other, similar measurements; the capacity of the tool to discriminate between different types of Scratch projects; as well as the vision and feelings of educators who are using the tool in their lessons. The paper also highlights the actions that are still pending to complete the formal validation of Dr. Scratch.

1 Introduction

`Dr. Scratch` [MLRRG15] is a free/libre/open source tool that analyzes `Scratch` [RMMH+09] projects to assess the level of development of computational thinking (CT) skills by inspecting the source code of the programs. `Dr. Scratch` is inspired by previous assessment tools for visual programming languages, such as `Scrape` [WHT11], `Fairy Assessment` [WDCK12] or `REACT` [KBNR14], and is based on `Hairball` [BHL+13], a static code analyzer for `Scratch` projects that detects potential issues in the code [FCB+13].

Since the `Hairball` architecture is based on plug-ins, we developed new plug-ins to detect several bad programming habits or *bad smells* that educators frequently detect in their work with middle and high school students [MR14]. In addition, we created a web-based service that facilitates the analysis of the projects and provides feedback with ideas to improve the code. This web-tool is what we called `Dr. Scratch`.

In this paper we present the current state of the validation process of `Dr. Scratch`, highlighting the actions that have already been performed or are under development, and pointing to the activities that are still pending to complete its formal validation.

2 Dr. Scratch CT analysis

The main feature of `Dr. Scratch` is the analysis of CT skills. Aiming to come up with a CT score system, we reviewed prior work proposing theories and methods for assessing the development of programming and CT skills of learners [WHC12, SF13, BR12] and collaborated with educators with years of experience using `Scratch` in their lessons. Table 1 summarizes the CT assessment, which is based on the degree of development of seven dimensions of the CT competence: abstraction and problem decomposition, logical thinking, synchronization, parallelism, algorithmic notions of control flow, user interactivity and data represen-

Proceedings of the Seminar Series on Advanced Techniques and Tools for Software Evolution SATToSE 2017 (sattose.org).
07-09 June 2017, Madrid, Spain.

tation. These dimensions are statically evaluated by inspecting the source code of the analyzed project and given a score from 0 to 3, resulting in a total mastery score that ranges from 0 to 21 when all seven dimensions are aggregated. With this information the tool generates a feedback report that is displayed to learners, as shown in Figure 1.

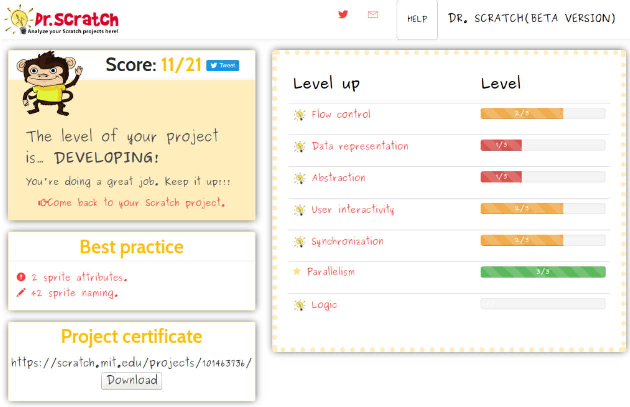

Figure 1: Dr. Scratch Analysis Result for *Star Wars StickMan DressUp*, a Scratch project available at https://scratch.mit.edu/projects/101463736/.

3 Dr. Scratch Validation Process

Different actions have been performed, or are under way, to validate `Dr. Scratch` from different perspectives. Firstly, we want young students to be able to analyze their projects and independently learn from the tips that the tool provides, so we organized a series of workshops in schools to check the extent to which learners improve their coding skills while using the tool in real life scenarios (ecological validity, section 3.1). In addition, the CT score provided by the tool is, to some degree, similar to other measurements, such as teacher grades or software engineering complexity metrics, and the score has therefore been compared with them to study relationships and correlations (convergent validity, section 3.2). On other hand, `Scratch` creations are categorized under different types of projects, and it is interesting to find out if this topology is replicated when projects are analyzed with `Dr. Scratch` (discriminant validity, section 3.3). Lastly, the tool will not be used unless educators feel that it measures what it promises, so we have surveyed several hundreds teachers on this regard (face validity, section 3.4).

A key validation action that is still pending is the study of a big number of analyses to identify potential clusters of CT dimensions assessed by the tool (factorial validity, section 3.5), which could lead to the grouping of some dimensions, thus simplifying the feedback provided to users.

3.1 Ecological Validity

One of the main objectives of `Dr. Scratch` is to encourage students to improve their programming skills. Aiming to check its effectiveness regarding this goal, we organized a series of workshops with over 100 students in the range from 10 to 14 years in 8 different schools [MLRRG15]. During the workshops, participants analyzed one of their projects with `Dr. Scratch`, read the feedback report provided by the tool, and tried to improve their projects using the tips included in the report. After 1 hour of working with `Dr. Scratch`, students increased their CT score and improved their programming skills, which proves that the tool is useful for learners of these ages.

Nevertheless, even though `Dr. Scratch` offers a CT-dependent feedback report, the workshops showed that the tool was especially useful for older students (12-14 years old) with an intermediate initial CT score. Future interventions will help us modify the feedback report so that younger students as well as learners with basic or advanced CT levels can also make the most out of the tool.

3.2 Convergent Validity

A big step in the validation process is the comparison of the evaluations provided by `Dr. Scratch` with other measurements of similar constructs. We identified three assessments that could be used in this regard: the ones provided by human experts, software engineering complexity metrics and the `CT-test` [RGPGJF16], a 28-item multiple-choice test aimed at 12-13 years old students to assess their CT skills.

On one hand, in order to compare the automatic scores provided by `Dr. Scratch` with the evaluations by human experts, we organized a programming contest for primary and secondary students [MLRGHR17]. This allowed us to gather and study over 450 evaluations of Scratch projects given by 16 experts in computer science education. The results show strong correlations between automatic and manual evaluations, which could be considered as a validation of the metrics used by the tool. According to the assessment research literature [CH12], the tool is *ideally convergent* with expert evaluators, since the correlation detected between measurements is greater than $r = .70$.

On other hand, we compared the `Dr. Scratch` scores with two classic software engineering metrics that are globally recognized as a valid measurement for the complexity of a software system: Mc-

Table 1: Level of Development for Each CT Dimension Assessed by Dr. Scratch [MLRRG15]

CT dimension	**Basic**	**Intermediate**	**Proficient**
Logical Thinking	if	if else	logic operations
Data representation	modifiers of object properties	variables	lists
User interactivity	green flag	keyboard, mouse, ask and wait	webcam, input sound
Control flow	sequence of blocks	repeat, forever	repeat until
Abstraction and problem decomposition	more than one script	use of custom blocks	use of 'clones' (instances of sprites)
Parallelism	two scripts on green flag	two scripts on key pressed or sprite clicked	two scripts on receive message, video/audio input, backdrop change
Synchronization	wait	message broadcast, stop all, stop program	wait until, when backdrop changes to, broadcast and wait

Cabe's Cyclomatic Complexity and Halstead's metrics [MLRRG16a]. By comparing the results after analyzing 95 `Scratch` projects, we found positive, significant, moderate to strong correlations between measurements, which could also be considered as a validation of the complexity assessment process of the tool.

Finally, in an intervention involving an 8-weeks programming course with `Scratch` in three Spanish middle schools, with a total sample of n=71 students, we compared the `Dr. Scratch` scores with the results of the `CT-test`. The findings show a positive, moderate, and statistically significant correlation between tools, both in predictive and concurrent terms [RGMLR17]. As we expected, the convergence is not total since, although both tools are assessing the same psychological construct, they do it from different perspectives: summative-aptitudinal (`CT-test`) and formative-iterative (`Dr. Scratch`).

3.3 Discriminant Validity

Projects shared in the `Scratch` repository are categorized under one or more project types, being the most common games, animations, music, art and stories, although there are other categories such as tutorials or simulations. Aiming to check whether `Dr. Scratch` is able to detect differences in the CT dimensions developed when programming different types of `Scratch` projects, we randomly downloaded 500 projects from the main categories in the repository. Although this work has not been published (at the moment of writing this paper it is still under revision), the results of a K-means cluster analysis confirm that different types of projects can be used to develop distinct CT dimensions, and consequently, the existing typology of `Scratch` projects is empirically replicated when the projects are subjected to the `Dr. Scratch` mastery score. For instance, while *if* and *if else* instructions are commonly present at games, this is not the case for stories, which usually have a linear structure without branches. Therefore, games tend to score high in the `Dr. Scratch` logical thinking dimension, while the opposite is true for stories.

3.4 Face Validity

Another main goal of the tool is to support teachers in the evaluation tasks. It is thus important that educators feel that `Dr. Scratch` does what it claims to do (i.e. assessing CT). Pursuing this goal we prepared a survey for teachers who participate in a 40-hours `Scratch` coding training course. At the moment of writing this paper more than 320 educators have submitted the survey and another 120 are taking the course. The partial results indicate that 84% of participating teachers feel that the CT analysis of the tool is measured in a correct way. Having educators who teach at different grades and who have reached distinct levels of development of CT skills during the course will allow the study of potential differences in their opinions about the usefulness of the tool based on these variables.

3.5 Factorial Validity

A key pending validation action is to study the relationships between the different CT dimensions assessed by `Dr. Scratch` aiming to identify clusters of dimensions that share sufficient variation. This factorial analysis, if performed on a big enough number of projects, could lead to the grouping of some of the dimensions, which would therefore simplify the feedback report that the tool displays to learners.

For this action we will use the dataset made available from [AH16], which consists of 250,166 projects scraped from the Scratch repository.

4 Conclusions

This paper presents the current state of the validation process of Dr. Scratch, a tool that enables analyzing Scratch projects to assess the development of CT skills. Most of the investigations required to validate the tool are already done and published (ecological and convergent validity), or are in progress and will soon be published (discriminant and face validity). In consequence, even though there is also a key action that is still pending (factorial validity), the verification process of Dr. Scratch is close to be finalized.

It must be noted, nonetheless, that the analysis of just one project cannot provide a full picture of a learner's CT skills, as there are perfectly valid simple projects that do not require any modifications to include more complex structures (those that give a higher CT score). Consequently, we plan to add a new feature to enable the creation of user accounts, aiming that the analysis of the portfolio of projects of a learner will provide a richer, more accurate picture.

Furthermore, this new feature will also offer the possibility of easily tracking learners progression and projects evolution, both in terms of software complexity and presence of *bad smells*. The following papers can illustrate the kind of investigations in which the tool could be used in this regard: i) A study on the relationship between socialization and coding skills [MLRRG16b] that made use of the whole Scratch repository from 2007 to 2012 [HMH17], wherein we used an adaptation of Dr. Scratch to assess the sophistication of more than 1.5 million projects authored by almost 70,000 users. ii) An investigation on the presence of copy-and-paste in Scratch projects, in which we correlated the assessment of the CT skills of learners, measured by Dr. Scratch, with the existence of software clones in over 230,000 projects [RMLAH17].

Acknowledgements

This work has been funded in part by the Region of Madrid under project "eMadrid - Investigación y Desarrollo de tecnologías para el e-learning en la Comunidad de Madrid" (S2013/ICE-2715).

References

[AH16] Efthimia Aivaloglou and Felienne Hermans. How kids code and how we know: An exploratory study on the Scratch repository. In *Proceedings of the 2016 ACM Conference on International Computing Education Research*, pages 53–61. ACM, 2016.

[BHL+13] Bryce Boe, Charlotte Hill, Michelle Len, Greg Dreschler, Phillip Conrad, and Diana Franklin. Hairball: Lint-inspired static analysis of Scratch projects. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, pages 215–220, New York, NY, USA, 2013. ACM.

[BR12] Karen Brennan and Mitchel Resnick. New frameworks for studying and assessing the development of Computational Thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada*, pages 1–25, 2012.

[CH12] Kevin D Carlson and Andrew O Herdman. Understanding the impact of convergent validity on research results. *Organizational Research Methods*, 15(1):17–32, 2012.

[FCB+13] Diana Franklin, Phillip Conrad, Bryce Boe, Katy Nilsen, Charlotte Hill, Michelle Len, Greg Dreschler, Gerardo Aldana, Paulo Almeida-Tanaka, Brynn Kiefer, Chelsea Laird, Felicia Lopez, Christine Pham, Jessica Suarez, and Robert Waite. Assessment of computer science learning in a Scratch-based outreach program. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, pages 371–376, New York, NY, USA, 2013. ACM.

[HMH17] Benjamin Mako Hill and Andrés Monroy-Hernández. A longitudinal dataset of five years of public activity in the scratch online community. *Scientific Data*, 4, 2017.

[KBNR14] Kyu Han Koh, Ashok Basawapatna, Hilarie Nickerson, and Alexander Repenning. Real time assessment of computational thinking. In *Visual Languages and Human-Centric Computing (VL/HCC), 2014 IEEE Symposium on*, pages 49–52. IEEE, 2014.

[MLRGHR17] Jesús Moreno-León, Marcos Román-González, Casper Harteveld, and Gregorio Robles. On the automatic assessment of computational thinking skills: A comparison with human experts. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, pages 2788–2795, New York, NY, USA, 2017. ACM.

[MLRRG15] Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. Dr. Scratch: Automatic analysis of scratch projects to assess and foster Computational Thinking. *RED. Revista de Educación a Distancia*, 15(46), 2015.

[MLRRG16a] Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. Comparing computational thinking development assessment scores with software complexity metrics. In *2016 IEEE Global Engineering Education Conference (EDUCON)*, pages 1040–1045, April 2016.

[MLRRG16b] Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. Examining the relationship between socialization and improved software development skills in the scratch code learning environment. *Journal of Universal Computer Science*, 22(12):1533–1557, 2016.

[MR14] Jesús Moreno and Gregorio Robles. Automatic detection of bad programming habits in Scratch: A preliminary study. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, pages 1–4, Oct 2014.

[RGMLR17] Marcos Román-González, Jesús Moreno-León, and Gregorio Robles. Complementary tools for computational thinking assessment. In *CTE 2017: International Conference on Computational Thinking Education 2017*, page In press, July 2017.

[RGPGJF16] Marcos Román-González, Juan-Carlos Pérez-González, and Carmen Jiménez-Fernández. Which cognitive abilities underlie computational thinking? criterion validity of the computational thinking test. *Computers in Human Behavior*, pages –, 2016.

[RMLAH17] G. Robles, J. Moreno-León, E. Aivaloglou, and F. Hermans. Software clones in scratch projects: on the presence of copy-and-paste in computational thinking learning. In *2017 IEEE 11th International Workshop on Software Clones (IWSC)*, pages 1–7, Feb 2017.

[RMMH+09] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. Scratch: Programming for all. *Commun. ACM*, 52(11):60–67, November 2009.

[SF13] Linda Seiter and Brendan Foreman. Modeling the learning progressions of computational thinking of primary grade students. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, ICER '13, pages 59–66, New York, NY, USA, 2013. ACM.

[WDCK12] Linda Werner, Jill Denner, Shannon Campe, and Damon Chizuru Kawamoto. The fairy performance assessment: Measuring computational thinking in middle school. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, pages 215–220, New York, NY, USA, 2012. ACM.

[WHC12] Amanda Wilson, Thomas Hainey, and Thomas Connolly. Evaluation of computer games developed by primary school children to gauge understanding of programming concepts. In *European Conference on Games Based Learning*, page 549. Academic Conferences International Limited, 2012.

[WHT11] Ursula Wolz, Christopher Hallberg, and Brett Taylor. Scrape: A tool for visualizing the code of Scratch programs. In *Poster presented at the 42nd ACM Technical Symposium on Computer Science Education*, Dallas, TX, 2011.